Oriented Boundary Graph: A Framework to Design and Implement 3D Segmentation Algorithms

Fabien Baldacci, Achille Braquelaire, Jean-Philippe Domenger
Université Bordeaux 1, CNRS, LaBRI, 351, cours de la Libération 33405 Talence cedex, France
{baldacci,achille,domenger}@labri.fr

Abstract

In this paper we show the interest of a topological model to represent 3D segmented image which is a good compromise between the complete but time consuming representations and the partial but not expressive enough ones. We show that this model, called Oriented Boundary Graph, provides an effective framework for both volumic image analysis and segmentation. The Oriented Boundary Graph provides an efficient implementation of a set of primitives suitable for the design complex segmentation algorithms and to implement the computation of the segmented image characteristics needed by such algorithms. We first present the framework and give the time complexity of its main primitives. Then, we give some examples of the use of this framework in order to efficiently design non-trivial image analysis operations and image segmentation algorithms. Those examples are applied on 3D CT-scan data.

1. Introduction

Designing generic and efficient segmentation algorithms remains a difficult task. By essence, the segmentation process is intrinsic to the area of interest and the nature of the class of images to be analyzed. This leads to a vast number of dedicated algorithms, each one being specific to a specific purpose. To solve this problem, a lot of research has been done in the field of image structuring. Indeed, structuring an image allows an efficient retrieval of geometrical features (such as lengths or curvatures) and/or topological features (such as neighborhood or Betti numbers) [5]. These features are both used as criteria to drive the segmentation process and to analyze its results. Furthermore, a topological structuring encodes neighboring information such as the adjacency relation between regions and provides a convenient layout for designing split and merge algorithms.

Existing models [7, 4] are either highly time and space consuming and for now cannot be efficiently used for real 3D segmentation problems, or does not provide enough features to allow to design complex segmentation algorithm. To overcome this problem, we developed a new structuring model called the Oriented Boundary Graph [1]. Whereas this model does not explicit the whole topology of a segmented image such as a topological map based representation does, it is designed to reduce both time and memory complexity while providing primitives and features for efficient implementations of split and merge algorithms. In order to fulfill these requirements, a kernel of primitives has been implemented with an API dedicated to image segmentation. Using this set of primitives that provides an efficient extraction of main topological and geometrical features, a large collection of segmentation algorithms can be designed and optimized for specific image processing applications.

In this article we present some representative primitives composing our framework and detail the time complexity obtained by using the Oriented Boundary Graph model for each of them. We give several examples of use of this framework in order to:

- implement existing methods such as graph-based image segmentation operations [2],
- redefine some operations such as morphological filters for them to be more effective,
- design non-trivial segmentation algorithms.

In Section 2 we present the OBG model and the related framework, and in Section 3 we give some examples of how to use this framework. Examples are applied on 3D images obtained by CT-scan.
2. Framework

2.1. Oriented Boundary Graph Model

Let us first describe the model on which the proposed framework is based. This model, called Oriented Boundary Graph (OBG), provides a topological representation of a segmented 3D image, associated with a geometrical representation called Boundary Image. The Boundary Image encodes elements of the decomposition of the 3D discrete space into elements of dimension 3, 2, 1 and 0, respectively called voxels, surfels, linels and pointels (Fig. 1) [6]. This decomposition leads to an intervoxel representation of the segmented image geometry where regions are made of voxels, and surfaces between regions are made of surfels. Surfaces can be decomposed in a canonical way into surface elements, called S-Patches, according to their incident regions. A S-Patch is a maximum set of connected surfels belonging to the boundary shared by two regions. A partition both defines a set of regions and a set of S-Patches (Fig. 2(a)) from which the OBG can be extracted (Fig. 2(b)). For further details, we encourage to read [1].

![Figure 1. Elements of the decomposition of 3D discrete space.](image1)

![Figure 2. Example of configuration](image2)

The OBG associates a node to each region of the partition and an edge to each S-Patch. It is defined in order to include all the elements required to implement efficiently the primitives of our framework. The result is an hyper-graph which allows:

- an efficient access to any element of the partition,
- to efficiently retrieve any geometrical element from its topological corresponding element, and conversely,
- to efficiently update the model to take into account partition modifications.

2.2. Provided Framework

The OBG framework is made of three kinds of functions: the ones dealing with topological elements, the ones dealing with geometrical elements, and the ones updating the OBG according to the modifications of the image partition. A similar framework was defined and validated in 2D as a set of primitives suitable for an environment of segmentation of 2D images [3].

Let us give some examples of basic primitives composing our framework. Time complexities of those functions are summarized in Tab. 1. We denote the number of voxels of \( r \) by \( |r|_v \), the number of surfels composing the boundary of \( r \) by \( |\partial r|_s \), the number of surfels composing the S-Patch \( S \) by \( |S|_s \) and the number of S-Patches composing the boundary of \( r \) by \( |\partial r|_{SP} \).

The two main update functions are the split and the merge functions. The split function updates the OBG according to a new partition associated with the splitted region \( r \). The merge function updates the OBG such as two given adjacent regions \( r_1 \) and \( r_2 \) are merged into one. The OBG model also provides an efficient access to geometrical elements of a partition. The framework includes several primitives such as the region Boundary function which gives the list of surfels composing the boundary of a given region \( r \), and the common boundary function which computes the list of surfels belonging to the boundaries of two given regions \( r_1 \) and \( r_2 \). In a similar way, an efficient access to each topological relation (neighborhood, inclusion, etc) can be performed by using the primitives of our framework. For a given region, the function get neighbors computes the list of its neighbors, the function get s patches computes the list of S-Patches composing its boundary, and is isolated determines if this region corresponds to a cavity of another one.

Each function has a time complexity proportional to the number of cells composing the computed geometrical or topological element, leading to an efficient framework implementation.

3. From our Framework to Image Analysis

This framework allows to efficiently compute most of the information that could be of interest in an image analysis environment. We are going to give different kind of use of such a framework.
Table 1. Time complexity of some primitives of our framework.

<table>
<thead>
<tr>
<th>Function</th>
<th>Time Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>split</td>
<td>$O(</td>
</tr>
<tr>
<td>merge</td>
<td>$O(</td>
</tr>
<tr>
<td>region_boundary</td>
<td>$O(</td>
</tr>
<tr>
<td>common_boundary</td>
<td>$O(</td>
</tr>
<tr>
<td>get_neighbors</td>
<td>$O(</td>
</tr>
<tr>
<td>get_s_patches</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>is_isolated</td>
<td>$O(1)$</td>
</tr>
</tbody>
</table>

3.1. Topological distance between regions

Considering the OBG sub-graph only made of nodes corresponding to regions and edges corresponding to S-Patches, algorithms from the graph theory can be applied. For example, the topological distance between two regions can be defined as the length of the shortest path between their nodes.

The explicit encoding of nodes and edges allows to associate some values to each region and to each S-Patch (such as their size, mean value, mean gradient, etc.). Thanks to those data, it is possible to iterate some graph-cuts operations [2] which merge several regions at each step.

3.2. Morphological Filters

Since the proposed framework gives an efficient access to geometrical elements composing an image partition (voxels of regions, surfels of S-Patches, etc), usual image analysis operations can take advantage of the OBG in order to get an optimized implementation. For instance, surfaces processing can thus be applied only on necessary surfels and do not need to traverse the whole set of voxels of the image. As an example let us consider morphological filters applied on segmented images. This operation can be used in order to remove noise on surfaces such as illustrated on Fig. 3. This operation applied on an image made of $|I|$ voxels with a structural element made of $|D|$ voxels has a time complexity of $O(|I| \times |D|)$. On most of the image voxels, morphological filters have no effect. By using the region_boundary function on the processed region $r$ and according to the topological orientation (inside/outside) of the surface, it is possible to apply the structural element only upon voxels on which changes may occur. In the case of a dilatation, each voxel of the structural element has to be centered once on each voxel both incident to a surfel of $\partial r$ and belonging to $r$. Erosion can be adapted in a similar way. The time complexity of this optimized implementation is $O(|\partial r|_s \times |D|^2)$.

3.3. 3D Image Segmentation

The OBG model provides a framework allowing an efficient implementation of basic features extraction from a segmented image: region characteristics (size, mean, variance, etc.), surface characteristics (size, gradient, curvature, etc.) and topological characteristics (region neighborhood, surface neighborhood, inclusion relation). These basic features allow the design of high-level primitives, providing a wide range of segmentation criteria and image analysis functionalities. This primitive set composed our segmentation framework, from which it is simple to design a segmentation algorithm given a specific segmentation problem. It allows to test and mix the result of several methods and criteria, and thus, by tuning the criteria applied with the most appropriated method, to converge to an algorithm solving the segmentation problem.

In order to show how easy the design of segmentation algorithm is, we give several operation examples that can be parts of it. The first presented operation consists in removing all isolated regions sized under a given threshold from a partition. Indeed, those regions are usually considered as noise regions. The Algorithm 1 shows how to implement this operation with our framework leading to a time complexity proportional to the number of regions of the partition.

Algorithm 1 Delete Little Isolated Regions

Require: an OBG $G$, a maximum size $s$

1: regions = get_all_regions($G$)
2: for all Region $R \in$ regions do
3:   if is_isolated($R$) and size($R$) < $s$ then
4:     $r'$ = merge($R$.get_neighbor($R$))
5:     regions.append($r'$)
6: end if
7: end for

The second example (Algorithm 2) consists in merging a given region with its neighbor forming the best
pair according to a given criteria. The time complexity only depends on the number of neighbors $|\mathcal{V}_r|$ and on the criteria time complexity $O(C)$ for a whole complexity of $O(|\mathcal{V}_r| \times O(C))$.

**Algorithm 2 Delete**

Require: an OBG $G$, a region $r$, a criteria $C$, a threshold $t$

1: for all Region $r' \in$ get_neighbors($r$) do
2: if $C(r, r') > t$ then
3: $t = C(r, r'), r'' = r'$
4: end if
5: end for
6: merge($r, r''$)

The last presented operation (Algorithm 3) is a region growing algorithm that iterates the previous algorithm on each neighbor of a set of regions of interest. Those neighbors are not necessarily merged with the region of interest. The list of regions of interest can be extracted by traversing the OBG graph, only retaining regions satisfying criteria.

**Algorithm 3 Region Growing**

Require: an OBG $G$, a list of region $l$, a criteria $C$.

1: repeat
2: for all Region $R \in l$ do
3: for all $R' \in$ get_neighbors($R$) do
4: Delete($R', C$)
5: end for
6: end for
7: until $G$ remains unchanged

A whole segmentation process was defined combining the previous algorithms. The criteria used is a combination of the mean difference of regions, the length and the mean gradient of common boundaries. This algorithm was applied on CT-scan data of human hip area, giving promising results for the segmentation of the immature coxal bone (Fig. 4). The whole segmentation process applied on images with size from $512 \times 512 \times 203$ to $512 \times 512 \times 342$ voxels, takes from 20 to 45 seconds on an Intel Core 2 Duo at 2.66 GHz computer. These results will be presented in a future work.

4. Conclusion

In this article we presented the framework provided by the Oriented Boundary Graph structuring model. We showed that this model is a good compromise between full topological representation like topological maps which are too time and space consuming for real cases of 3D images segmentation, and basic models like region labelling which are not expressive enough to allow to design real algorithms. This model is thus convenient to develop the primitives required by a 3D image analysis platform. This set of primitives highly facilitates the design and the test of high level segmentation algorithm to solve concrete problems. The full comparison of the Oriented Boundary Graph model with the Topological Map one and the Region Adjacency Graph one is the object of a next longer article. It will also include more examples of use of our model. We plan to develop a fully functional environment integrating both classical and newly defined segmentation and analysis operations developed with the OBG. This platform will be available on our website.

**Acknowledgments.** The authors wish to thank Pr. J.-F. Chatell, Director of the Radiology Department in Pellegrin’s Children Hospital, and H. Coqueugniot, CNRS Researcher from PACEA, for access to CT scan data.

**References**


